Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 1 (2012) 229–237

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Computational Science, ICCS 2010

Efficient design of exponential-Krylov integrators for large scale computing

M. Tokman[a,*], J. Loffeld[a]

[a]*School of Natural Sciences, University of California, 5200 N. Lake Road, Merced, CA 95343*

Abstract

As a result of recent resurgence of interest in exponential integrators a number of such methods have been introduced in the literature. However, questions of what constitutes an efficient exponential method and how these techniques compare with commonly used schemes remain to be fully investigated. In this paper we consider exponential-Krylov integrators in the context of large scale applications and discuss what design principles need to be considered in construction of an efficient method of this type. Since the Krylov projections constitute the primary computational cost of an exponential integrator we demonstrate how an exponential-Krylov method can be structured to minimize the total number of Krylov projections per time step and the number of Krylov vectors each of the projections requires. We present numerical experiments that validate and illustrate these arguments. In addition, we compare exponential methods with commonly used implicit schemes to demonstrate their competitiveness.

Keywords: exponential integrators, Krylov projections, stiff systems, large scale computing

2000 MSC: 65F10, 65F60, 65L04, 65L05, 65M22

1. Introduction

While the first exponential time integrators were introduced back in the 1960's [1, 2, 3] their popularity among numerical analysts and practitioners has been limited. Initially the main reason for such underutilization was the high computational cost of these schemes. Solving systems of ODEs with an exponential method requires evaluation of a product of an exponential or exponential-type functions of a large matrix with a vector. Even for moderately-sized systems this operation becomes prohibitively expensive if standard techniques such as Taylor or Pade approximations are employed [4]. However, a proposal to use a Krylov projection algorithm for this task significantly reduced computational cost. This idea first appeared in a paper by Nauts and Wyatt [5] where they used Krylov projection to compute exponentials of symmetric matrices that represented discrete Hamiltonian operators, and was later used by Park and Light [6] to exponentially propagate the Schrödinger equation. Van der Vorst extended this idea and proposed to apply Krylov projection to approximate general functions of matrices [7]. A resurgence of interest in exponential methods followed these ideas and a number of such methods have been proposed in the last decade [8, 9, 10, 11, 12].

*Corresponding author
Email address: mtokman@ucmerced.edu (M. Tokman)

1877-0509
doi:10.1016/j.procs.2010.04.026

Coupling exponential methods with the Krylov projection algorithm makes these time integrators much more appealing for large scale computing. Still many questions remain to be answered to enable wide application of these methods to scientific problems. In particular, thorough performance comparisons with state-of-art explicit and implicit integrators are needed and it remains to be demonstrated how details of the design of an exponential integrator affect its performance. This paper presents some results pertaining to the former question and focuses on the latter issue. We consider exponential integrators as methods that can allow for significant computational savings in integrating large stiff systems of ODEs and from that perspective discuss what constitutes an optimal design of an exponential integrator. The paper is organized as follows. Section 2 provides an overview of exponential methods for general nonlinear systems of ODEs and outlines the main features that influence performance of an exponential-Krylov integrator. A suite of test problems is presented in Section 3 and the ideas of previous sections are illustrated with numerical examples. Finally, conclusions and directions for future study are presented in Section 4.

2. Structure of exponential integrators

2.1. General derivation and important construction considerations for exponential integrators

In order to illustrate what choices have to be made in the design of an exponential integrator we begin by presenting the general derivation of such schemes. Consider the initial value problem for an autonomous nonlinear system of ODEs

$$y' = f(y), \quad y(t_0) = y_0, \tag{1}$$

where $y \in \mathbb{R}^N$. There is no loss of generality in considering an autonomous system since a non-autonomous one can always be converted to the autonomous form by adding the equation $t' = 1$. If the first-order Taylor expansion of $f(y)$ around y_0 exists we can re-write Eq. (1) as

$$y' = f(y_0) + f'(y_0)(y - y_0) + r(y) \tag{2}$$

with the nonlinear remainder of the first-order Taylor expansion denoted as $r(y) = f(y) - f(y_0) - f'(y_0)(y - y_0)$ and the Jacobian matrix $f'(y_0) \in \mathbb{R}^{N\times N}$. Using the integrating factor $e^{-f'(y_0)t}$ we can find the integral form of the solution to this system at time $t_0 + h$ as

$$y(t_0 + h) = y_0 + \frac{e^{f'(y_0)h} - I}{hf'(y_0)} hf(y_0) + \int_{t_0}^{t_0+h} e^{f'(y_0)(t-t_0)} r(y(t))dt. \tag{3}$$

After setting $A_0 = f'(y_0)$ and changing the integration variable to $s = (t - t_0)/h$ in Eq. (3) we obtain

$$y(t_0 + h) = y_0 + \frac{e^{hA_0} - I}{hA_0} hf(y_0) + \int_0^1 e^{hAs} hr(y(s))ds. \tag{4}$$

Equation (4) serves as a starting point in derivation of an exponential method. Alternative derivations are also available, particularly when the nonlinearity is decomposed into the linear and nonlinear terms as $f(y) = Ly + N(y)$ (see [13] for a brief history of exponential methods for such semi-linear problems). However for the general nonlinear systems of type (1) which are the focus of this paper, equation (4) is a convenient starting point for deriving existing exponential methods by interpreting t_0 as the latest time where an approximate solution is available, considering h as an integration step size and approximating the solution $y(t_0 + h)$.

Constructing an exponential integrator using (4) requires accomplishing two tasks: (I) developing an approximation to the nonlinear integral $\int_0^1 e^{hAs} hr(y(s))ds$ and (II) building an algorithm to evaluate products of functions of matrices and vectors arising from the second term of the right-hand-side of (4) and possibly from the approximation chosen for the integral in (I). For example, task (I) can be accomplished by approximating the nonlinear integral using the Runge-Kutta approach. With a two-stage Runge-Kutta-type approximation we can construct the two-stage exponential Runge-Kutta schemes [11]:

$$r_1 = y_0 + a_{11}\varphi_1(\gamma_{11}hA)hf(y_0), \tag{5}$$

$$y_1 = y_0 + a_{21}\varphi_1(\gamma_{21}hA)hf(y_0) + a_{22}\varphi_2(\gamma_{22}hA)r(r_1), \tag{6}$$

where y_1 is an approximation to the solution $y(t_0 + h)$, $\varphi_1(z) = \frac{e^z-1}{z}$, and $\varphi_2(z) = \frac{e^z-1-z}{z^2}$. Choosing $a_{11} = a_{21} = \gamma_{21} = \gamma_{22} = 1$, $\gamma_{11} = 1/2$ and $a_{22} = 2/3$ yields the third-order exponential Runge-Kutta method EPIRK3 proposed in [11]. In general, a polynomial approximation to the nonlinear remainder function $r(y)$ in (4) will result in an exponential scheme which approximates the solution as a linear combination of the products of type $\varphi_k(\gamma hA)v_k$ with $v \in \mathbb{R}^N$ and functions $\varphi_k(z)$ defined as

$$\varphi_k(z) = \int_0^1 e^{z(1-s)} \frac{s^{k-1}}{(k-1)!} ds, \qquad k = 0, 1, 2, ... \tag{7}$$

Obviously either Runge-Kutta or multistep approaches can be used in the derivation as well as any other construct that yields an approximation to the integral in (4). Once a certain ansatz for the approximation to the solution as a linear combinations of terms $a_{lk}\varphi_k(\gamma_{lk}hA)v_{lk}$ is assumed, the order conditions for the coefficients a_{lk}, γ_{lk} can be derived and solved to obtain exponential integrators of the desired order.

After constructing an exponential integrator one needs to address task (II) and to choose an algorithm to approximate the products of functions $\varphi_k(\gamma hA)$ and vectors v_{lk}. For small systems a number of techniques such as Taylor or Pade expansions can be used [4]. If the system size N is large, Krylov projection algorithm becomes the method of choice [14]. Thus a product of a function of a matrix $g(A)$ and a vector v is approximated using projection of the matrix and the vector onto the Krylov subspace $K_m(A, v) = span\{v, Av, ..., A^{m-1}v\}$ as follows. The orthonormal basis $\{v_1, v_2, ..., v_m\}$ of $K_m(A, v)$ is constructed using the modified Arnoldi iteration [15, 14] which can be written in matrix form as

$$AV_m = V_m H_m + h_{m+1,m} v_{m+1} e_m^T \tag{8}$$

where $e_m = (0, \ldots, 0, 1, 0, \ldots, 0)^T$ is the unit vector with 1 as the mth coordinate, $\{v_1, v_2, ..., v_m, v_{m+1}\}$ is an orthonormal basis of $K_m(A, b)$, $V_m = [v_1\ v_2\ ...\ v_m] \in \mathbb{R}^{N\times m}$, and

$$H_m = V_m^T A V_m \tag{9}$$

is an upper Hessenberg matrix calculated as a side product of the iteration. Matrix $P = V_m V_m^T$ is a projector onto $K_m(A, v)$, thus $g(A)v$ is approximated as a projection

$$g(A)b \approx V_m V_m^T g(A) V_m V_m^T b. \tag{10}$$

Recalling Eq. (9) and observing that $v_1 = v/\|v\|_2$ we make the final approximation

$$g(A)v \approx \|v\|_2 V_m g(H_m) e_1. \tag{11}$$

This algorithm can be used to approximate any of the matrix-function vector products $g(A)v$ with $g(z) = \varphi_k(z)$. It is important to note that the Arnoldi iteration is scale invariant, i.e. once H_m and V_m are calculated for a particular matrix A and vector v, in order to calculate corresponding matrices for γA and v we simply need to scale H_m and V_m by the factor γ, the orthonormal basis does not have to be recalculated from scratch. If $\gamma > 1$ additional Krylov vectors might have to be added to achieve the desired accuracy, if $\gamma < 1$ the approximation with m Krylov vectors will be sufficient. The key to efficiency of the Krylov projection algorithm is keeping the size of the Krylov basis m small so that calculating $g(H_m)$ is cheap and can be done using Pade or any other approximation effective for small matrices. The Krylov basis size m is determined during the course of the iteration using appropriate residuals [16, 9]. Note that m will depend on the eigenvalues of the matrix A, the magnitude of the vector v and the type of function $g(z)$. It has been demonstrated analytically for matrices with a specific spectrum [17] and numerically for some matrices [11] that as k is increased the number of Krylov vectors m required to approximate the product $\varphi_k(A)v$ decreases.

It is clear that approximation of the products $\phi_k(h\gamma_{kl}A_0)v_{lk}$ will constitute the main cost of an exponential scheme since the rest of the required operations is limited to several vector additions and scalar-vector multiplications. Considering efficiency of the complete exponential-Krylov integrator from the perspective of tasks (I) and (II) it is clear that the computational cost of applying an exponential scheme to integration of a large system (1) depends on two main features of the chosen method: (i) the total number of products $\phi_k(h\gamma_{kl}A_0)v_{lk}$ that have to be computed and (ii) the number of Krylov vectors that each of these products will require to achieve prescribed accuracy. Thus if we want to construct an exponential integrator of a certain order it is prudent to derive a scheme which minimizes both of these parameters, i.e. requires the minimum possible number of Krylov projections and chooses appropriate functions $g(z)$ and small vectors v so that these projections are fast. Below we consider existing exponential integrators from this point of view and demonstrate how design affects their performance.

2.2. Comparing designs of exponential integrators

To illustrate how design affects performance of an exponential-Krylov integrator we consider several existing methods proposed for the solution of general large nonlinear systems (1). While the conclusions hold for methods of any order we choose to compare exponential integrators of order four. The first method, *Exp4*, has been developed by Hochbruck et al. [9] and is arguably the most widely known exponential integrator:

$$\begin{aligned}
&k_1 = \varphi_1(\tfrac{1}{3}hA_0)f(y_0), \quad k_2 = \varphi_1(\tfrac{2}{3}hA_0)f(y_0), \quad k_3 = \varphi_1(hA_0)f(y_0),\\
&w_4 = -\tfrac{7}{300}k_1 + \tfrac{97}{150}k_2 - \tfrac{37}{300}k_3, \quad u_4 = y_0 + hw_4, \quad r_4 = f(u_4) - f(y_0) - hA_0w_4,\\
&k_4 = \varphi_1(\tfrac{1}{3}hA_0)r_4, \quad k_5 = \varphi_1(\tfrac{2}{3}hA_0)r_4, \quad k_6 = \varphi_1(hA_0)r_4,\\
&w_7 = \tfrac{59}{300}k_1 - \tfrac{7}{75}k_2 + \tfrac{269}{300}k_3 + \tfrac{2}{3}(k_4 + k_5 + k_6), \quad u_7 = y_0 + hw_7, \quad r_7 = f(u_7) - f(y_0) - hA_0w_7,\\
&k_7 = \varphi_1(\tfrac{1}{3}hA_0)r_7,\\
&y_1 = y_0 + h(k_3 + k_4 - \tfrac{4}{3}k_5 + k_6 + \tfrac{1}{6}k_7).
\end{aligned} \tag{12}$$

Note that due to the invariance of the Arnoldi iteration discussed above, only three Krylov projections are needed - one to approximate k_1, k_2 and k_3, another to estimate k_4, k_5, k_6 and the third one to compute k_7. Each of these projections approximates a product of type $\phi_1(\gamma hA)v$. Note that the function $g(z) = \varphi_1(z)$ does not change, however we can expect that the vector v will decrease in magnitude from one Krylov projection to another if vectors u_i are better approximations to the solution as i increases.

The second method is an exponential propagation iterative Runge-Kutta (*EpiRK*) scheme proposed in [11]:

$$\begin{aligned}
&u_1 = y_0 + a_{11}h\varphi_1(\tfrac{1}{3}hA_0)f(y_0),\\
&u_2 = y_0 + a_{21}h\varphi_1(\tfrac{2}{3}hA_0)f(y_0) + a_{22}h\varphi_2(\tfrac{2}{3}hA_0)r(u_1),\\
&y_1 = y_0 + h\varphi_1(hA_0)f(y_0) + b_1h\varphi_2(hA_0)r(u_1) + b_2h[6\varphi_3(hA_0) - \varphi_2(hA_0)](-2r(u_1) + r(u_2)).
\end{aligned} \tag{13}$$

Several methods of third- and forth-order have been derived in [11], in particular, a fourth-order scheme *EpiRK4* with $a_{11} = 3/4, a_{21} = 3/4, a_{22} = 0, b_1 = 160/81, b_2 = 64/81$. The *EpiRK* methods are designed so that the following two principles hold. First, the number of Krylov projections required per time step is minimized by reusing the same vector v in the matrix-function-vector product at each new stage u_i, i.e. here only three Krylov projections must be executed with vectors v in the matrix-function-vector products being $f(y_0)$, $r(u_1)$ and $(-2r(u_1) + r(u_2))$. Second, the number of Krylov vectors each of these projections requires is minimized by having higher order $\varphi_k(z)$ functions that have to be approximated with each new Krylov projection, i.e. $\varphi_1(z)$ for the first projection, $\varphi_2(z)$ for second and $\psi_4(z) = 6\varphi_3(z) - \varphi_2(z)$ for the last Krylov projection.

The last scheme considered here is an exponential Rosenbruck-type scheme *ERow4-1* [18]

$$\begin{aligned}
s_1 &= \varphi_0(\tfrac{1}{2}hA_0)y_0 + \tfrac{1}{2}h\varphi_1(\tfrac{1}{2}A_0)g(y_0),\\
s_2 &= \varphi_0(\tfrac{1}{2}hA_0)y_0 + h\varphi_1(hA_0)g_n(s_1),\\
y_1 &= \varphi_0(hA_0)y_0 + h[\varphi_1(hA_0) - 14\varphi_3(hA_0) + 36\varphi_4(hA_0)]g(y_0)\\
&+ h[16\varphi_3(hA_0) - 48\varphi_4(hA_0)]g(s_1) + h[-2\varphi_3(hA_0) + 12\varphi_4(hA_0)]g(s_2),
\end{aligned} \tag{14}$$

with $g(y) = f(y) - A_0y$. In this formulation it appears that *ERow4-1* requires four Krylov projections since terms $\varphi_0(\gamma hA_0)y_0$ must be computed in addition to terms with vectors $g(y_0)$, $g(s_1)$ and $g(s_2)$. However, if we re-write this method in terms of $r(y)$ using the relation $r(y) = g(y) + f(y_0) - A_0y_0$ we obtain a different formulation of the method we call *ERow4-2*:

$$\begin{aligned}
&u_1 = y_0 + \frac{1}{2}h\varphi_1(\frac{1}{2}hA_0)f(y_0),\\
&u_2 = y_0 + h\varphi_1(hA_0)f(y_0) + h\varphi_1(hA_0)r(u_1),\\
&y_1 = y_0 + h\varphi_1(hA_0)f(y_0) + h[16\varphi_3(hA_0) - 48\varphi_4(hA_0)]r(u_1) + h[-2\varphi_3(hA_0) + 12\varphi_4(hA_0)]r(u_2)
\end{aligned} \tag{15}$$

In this form the method is similar to *EpiRK4* and requires only three Krylov projections per time step. Just as *EpiRK4*, *ERow4-2* uses higher order exponential functions, which we expect to result in faster Krylov convergence for subsequent projections.

3. Numerical experiments

In this section we demonstrate how the design of the exponential integrators impacts their performance. We compare constant time step implementations of the three exponential integrators in MATLAB. To illustrate competitiveness of these methods compared to commonly used integrators we include the *BDF4* scheme based on the backwards-differentiation formula of order four and the popular stiff integrator *RADAU5* [19]. For fair comparison both of these methods are implemented using the Krylov projection based algorithm GMRES to solve the linear systems within Newton iterations arising due to implicitness [20].

We have studied the performance of the methods using a suite of test problems (Allen-Cahn [21, 22], Burgers, Brusselator [23, 19], Gray-Scott [24], a semilinear parabolic equation [18], and a nonlinear diffusion equation NILIDI [25]), however for the sake of brevity we choose two representative systems to discuss here. The two-dimensional Allen-Cahn equation and the one-dimensional Burgers equation represent the two end points in the spectrum of problems we studied in terms of how quickly the Krylov projection iteration converges, i.e. to achieve prescribed accuracy the number of Krylov vectors needed per projection is on the order of tens for the Allen-Cahn equation while for the Burgers equation given the same tolerance this number is of the order of a hundred. For convenience we call the former problem "Krylov-easy" and the latter "Krylov-difficult". This terminology directly corresponds to a problem being less or more stiff. Note that all the tests were ran with the same prescribed tolerance for the Krylov projection residuals which was set to 10^{-12}, a value that is smaller then the accuracy requirement for the smallest time step size. Surely this means that the accuracy achieved for some of the Krylov iterations is excessive compared to practical tolerances for given step sizes but such an approach ensures consistent comparison across integrators and helps illustrate the general trends in their performance. Below we describe the two test problems and the parameter values used in the calculations.

Example 3.1: The two-dimensional Allen-Cahn equation

$$u_t = u - u^3 + \alpha \nabla^2 u, \quad x, y \in [0, 1] \tag{16}$$

with $\alpha = 0.1$ is complemented with the initial and Neumann boundary conditions given by $u = 0.4 + 0.1(x + y) + 0.1 \sin(10x) \sin(20y)$. The diffusive term is discretized with standard second-order finite differences and the problem is integrated over the time interval $t \in [0, 0.1]$.

Example 3.2: The one-dimensional Burgers equation

$$u_t = -uu_x + \nu u_{xx}, \quad x \in [0, 1] \tag{17}$$

with $\nu = 0.03$ and initial and Dirichlet boundary conditions prescribed using $u = (\sin(3\pi x))^3(1 - x)^{3/2}$. The diffusive term was discretized using the second-order centered finite-differences, the uu_x term was approximated as $uu_x \approx (u_{i+1}^2 - u_{i-1}^2)/(4\Delta x)$, $i = 1, ..., N$ and the problem was integrated over the time interval $t \in [0, 1]$.

Table 1 demonstrates how the number of Krylov vectors depends on the structure of an exponential integrator and the effect it has on the overall computational efficiency of the method. As anticipated since both *EpiRK4* and *ERow4-2* use the minimum number of three Krylov projections with the higher order $\varphi_k(z)$ functions for each, the number of Krylov vectors required per projection for these methods is smaller compared to *Exp4* and *ERow4-1*. This is reflected by the total CPU time spent by each of the methods to integrate the equations over the whole time interval (Tables 1). The importance of the reduction in Krylov iterations was particularly pronounced for the more demanding Burgers problem. As can be seen from Table 1(b) *EpiRK4* and *ERow4-2* required well less than half the CPU time of *Exp4* at coarse step sizes. Even at the finest step sizes, the savings offered by these two methods were still quite evident. The importance of using the higher order $\varphi_k(z)$ functions can be further illustrated by comparing performance of *ERow4-1* and *Exp4*. Despite the fact that *ERow4-1* has to compute one extra Krylov projection compared to other methods, it still manages to significantly outperform *Exp4* at coarse and medium step sizes. Since adding a vector to the Krylov basis requires orthonormalizing the new vector against every previously computed vector in the basis, the computational cost per vector goes up linearly with the basis size. Therefore the total cost of computing the Krylov basis increases quadratically with the basis size. Thus even modest reduction in the total number of the Krylov vectors per projection can result in significant CPU savings for large basis sizes. As can be seen by comparing with *EpiRK4* and *ERow4-2*, the savings are even greater when both the number of projections is reduced and the falloff of the number of Krylov vectors per projection happens more rapidly.

While the analysis above illustrates the effect of Krylov projections on the computational cost, in order to assess

the overall efficiency of a method the accuracy of the final approximation to the solution has to be taken into account. Precision diagrams displayed in Figure 1 show the relative performance of the integrators in terms of both accuracy and CPU time required. The problems were each run with several levels of resolution to show how the performance of a method scales with problem size. Figure 1 leads to the following conclusions about comparative performance of the methods. First, the effect of the Krylov iterations on efficiency becomes apparent particularly when a problem's stiffness is increased and it becomes more "Krylov-difficult", e.g. as the problem size grows for Allen-Cahn equation from $N = 50^2$ to $N = 150^2$ *EpiRK4* and *ERow4-2* become increasingly more efficient compared to other methods particularly for large step sizes h. Similar behavior can be read off the precision diagrams for the Burgers equation. For example, for step size $h = 0.1$ solution approximations for Burgers equation obtained by *Exp4* and *ERow4* have comparable accuracy, but relative CPU time of *ERow4-2* compared to *Exp4* improves from 60% to 40% as the problem size is increased from $N = 500$ to $N = 1500$. Second, we can see that as the stiffness of a problems is increased the bend in the precision curves particularly for large step sizes indicates that the problem becomes more "Krylov-difficult" and the relative computational cost of the methods becomes more pronounced (note the change in scale of separation between the curves). Note that the bend in the curves illustrates the importance of adaptivity in choosing the dimension of a Krylov subspace. The efficiency of the method is optimal if the tolerance for the residual of a Krylov iteration is calculated depending on the time step size. Our results on development of efficient adaptive algorithms are outside the scope of this paper and will be reported elsewhere. Finally, the figures make it apparent that the exponential methods compete very well with the standard implicit integrators. Note that some of the figures do not include *RADAU5*. The reason for that is the poor performance of this method for these values of h and N which puts it way off scale compared to the other schemes, i.e. the performance curve is so far to the right of the graph that we chose not to include it in the figures in order to preserve clarity in terms of relative performance for the rest of the schemes. In addition to overall computational savings that the exponential methods offer compared to *BDF4* and *RADAU5* we can also observe that the difficulty in Krylov convergence affects the implicit methods more severely compared to the exponential integrators. For example with the Allen-Cahn equation at step step $h = 0.01$, the CPU cost for *ERow4-2* compared to *BDF4* is about 74% for the smallest problem size. This gap increases to 51% for the largest problem size. The effect is similar but more pronounced for the Krylov-difficult Burgers equation. At $h = 0.01$, the CPU time ratio for *ERow4-2* compared to *BDF4* changes from 12% for the smallest problem size to about 5% for the largest problem size.

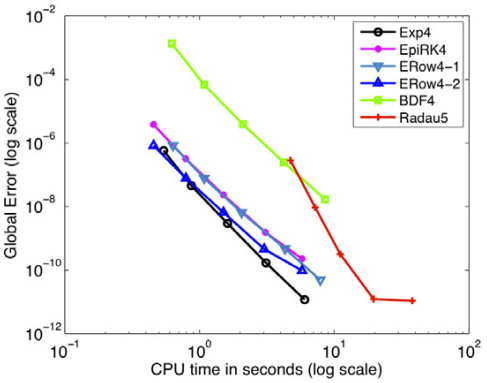
4. Conclusions and future work

In this paper we showed how the design of an exponential-Krylov integrator affects its performance. Specifically, we demonstrated that an integrator will be more efficient if it is designed to minimize the total number of Krylov projections per time step and the number of Krylov vectors that each of these projections requires. In addition, our studies reveal exponential-Krylov integrators as very competitive alternatives to more commonly used implicit schemes. More detailed studies of the comparative performance of the exponential and implicit schemes both with constant and adaptive time stepping will be presented elsewhere. In addition, we plan to explore the design principles outlined above to construct more optimized exponential-Krylov integrators and study their performance on large-scale scientific applications.

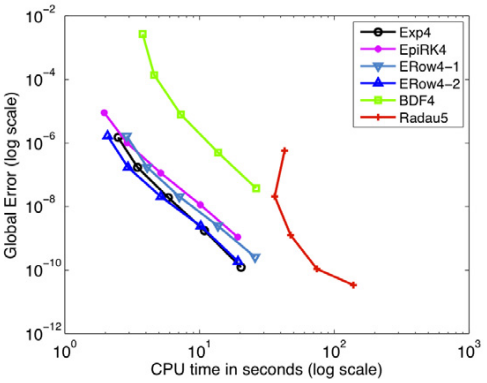
Acknowledgements

This work was supported in part by the NSF/DOE Partnership in Plasma Science grant #DMS-0317511 and a grant from the U.S. Department of Energy, Office of Science, Offices of Advanced Scientific Computing Research, and Biological & Environmental Research through the U.C. Merced Center for Computational Biology #DE-FG02-04ER25625.

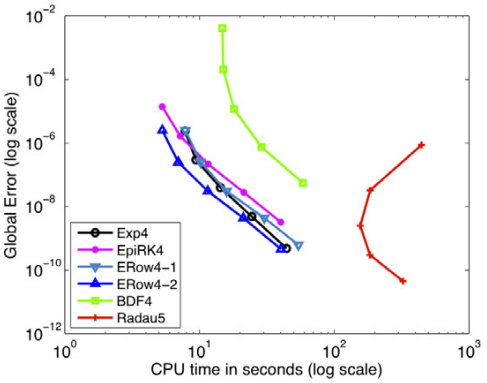
(a) Allen-Cahn with $N = 50^2$



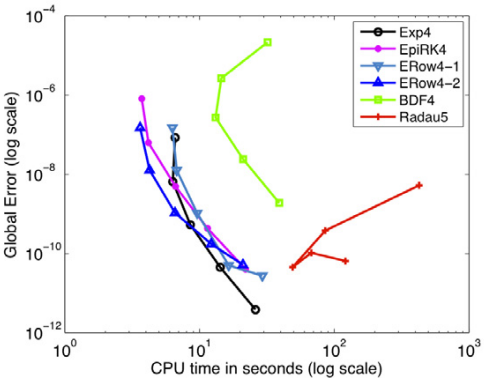
(b) Allen-Cahn with $N = 100^2$



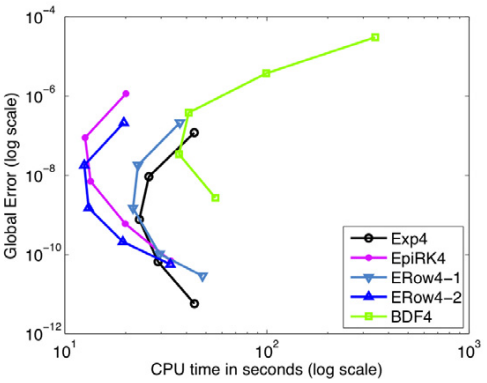
(c) Allen-Cahn with $N = 150^2$



(d) Burgers with $N = 500$



(e) Burgers with $N = 1000$



(f) Burgers with $N = 1500$

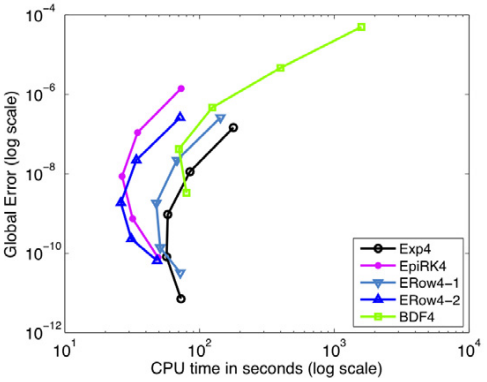


Figure 1: Precision diagrams for the Allen-Cahn (a-c) and Burgers (d-f) equations for $h = 0.01, 0.005, 0.0025, 0.00125, 0.000625$. Note that the axes scale changes from graph to graph.

Table 1: Average Krylov vectors counts and total CPU time.

(a) 2D Allen-Cahn with $N = 150^2$

$h = 0.01$	Average number of Krylov vectors				Total # of Krylov vectors	Total CPU time
	Projection 1	Proj. 2	Proj. 3	Proj. 4		
Exp4	32.0	25.8	26.7	n/a	84.5	2.48
EpiRK4	27.9	17.4	13.4	n/a	58.7	1.96
ERow4-1	28.8	23.7	23.5	17.1	93.1	2.80
ERow4-2	27.5	19.2	13.7	n/a	60.4	1.97
h = 0.005						
Exp4	20.6	16.2	17.1	n/a	53.8	3.65
EpiRK4	17.4	9.5	6.4	n/a	33.2	3.07
ERow4-1	18.5	14.1	14.1	9.7	56.4	4.30
ERow4-2	17.2	10.4	5.8	n/a	33.4	3.06
h = 0.0025						
Exp4	14.1	10.8	11.3	n/a	36.2	6.54
EpiRK4	11.4	4.6	3.3	n/a	19.4	5.49
ERow4-1	12.5	8.7	8.7	5.7	35.4	7.58
ERow4-2	11.4	5.5	3.3	n/a	20.2	5.44

(b) 1D Burgers with $N = 15000$

$h = 0.01$	Average number of Krylov vectors				Total # of Krylov vectors	Total CPU time
	Projection 1	Proj. 2	Proj. 3	Proj. 4		
Exp4	133.1	113.2	117.4	n/a	363.7	178.70
EpiRK4	116.7	64.2	42.5	n/a	223.4	73.05
ERow4-1	120.7	107.4	107.4	73.3	408.8	143.09
ERow4-2	114.8	73.3	33.4	n/a	221.5	71.78
h = 0.005						
Exp4	86.4	70.3	72.7	n/a	229.3	84.94
EpiRK4	74.0	32.6	19.7	n/a	126.2	34.71
ERow4-1	76.8	64.3	64.3	40.8	246.2	67.96
ERow4-2	72.9	36.7	14.7	n/a	124.2	33.98
h = 0.0025						
Exp4	57.7	44.1	46.2	n/a	147.9	58.12
EpiRK4	47.0	15.5	10.8	n/a	73.3	26.61
ERow4-1	49.8	37.5	37.5	23.7	148.5	47.82
ERow4-2	46.6	17.0	6.9	n/a	70.4	26.11

References

[1] J. Certaine, The solution of ordinary differential equations with large time constants, Wiley, 1967.
[2] D. A. Pope, An exponential method of numerical integration of ordinary differential equations, Comm. ACM 6 (1963) 491–493.
[3] J. Lawson, Generalized Runge-Kutta processes for stable systems with large Lipschitz constants, SIAM J. Numer. Anal. 4 (1967) 372–380.
[4] C. Moler, C. V. Loan, Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later, SIAM Rev. 45 (1) (2003) 3–49.
[5] A. Nauts, R. E. Wyatt, New approach to many-state quantum dynamics: the recursive-residue-generation method, Phys. Rev. Lett. 51 (1983) 2238–2241.
[6] T. J. Park, J. C. Light, Unitary quantum time evolution by iterative Lanczos reduction, J. Chem. Phys. 85 (1986) 5870–5876.
[7] H. A. V. der Vorst, An iterative solution method for solving $\mathbf{f}(\mathbf{a})\mathbf{x} = \mathbf{b}$ using Krylov subspace information obtained for the symmetric positive definite matrix $\mathbf{a}$, J. Comput. Appl. Math. 18 (1987) 249–263.
[8] R. A. Friesner, L. S. Tuckerman, B. C. Dornblaser, T. V. Russo, A method for exponential propagation of large systems of stiff nonlinear differential equations, J. Sci. Comput. 4 (1989) 327–354.
[9] M. Hochbruck, C. Lubich, H. Selhofer, Exponential integrators for large systems of differential equations, SIAM J. Sci. Comput. 19 (1998) 1552–1574.
[10] M. Hochbruck, A. Ostermann, Exponential integrators of rosenbrock-type, Oberwolfach Reports 3 (2006) 1107–1110.
[11] M. Tokman, Efficient integration of large stiff systems of ODEs with exponential propagation iterative (EPI) methods, J. Comp. Phys. 213 (2006) 748–776.
[12] A. Ostermann, M. Thalhammer, W. M. Wright, A class of explicit exponential general linear methods, BIT Num. Math. 46 (2) (2006) 409–431.
[13] B. Minchev, W. M. Wright, A review of exponential integrators for first order semi-linear problems, Technical Report (2005) 1–44.
[14] Y. Saad, Iterative methods for sparse linear systems, PWS Publishing Company, 1996.
[15] W. Arnoldi, The principle of minimized iteration in the solution of the matrix eigenvalue problem, Quart. Appl. Math. 9 (1951) 17–29.
[16] Y. Saad, Analysis of some Krylov subspace approximations to the matrix exponential operator, SIAM J. Numer. Anal. 29 (1992) 209–228.
[17] M. Hochbruck, C. Lubich, On Krylov subspace approximations to the matrix exponential operator, SIAM J. Numer. Anal. 34 (1997) 1911–1925.
[18] M. Hochbruck, A. Ostermann, J. Schweitzer, Exponential Rosenbrock-type methods, SIAM J. Numer. Anal. 47 (2009) 786–803.
[19] E. Hairer, G. Wanner, Solving Ordinary Differential Equations II: Stiff and Differential Algebraic Problems, 2nd Edition, Springer, 2004.
[20] D. Knoll, D. Keyes, Jacobian-free NewtonKrylov methods: a survey of approaches and applications, J. Comp. Phys. 193 (2004) 357397.
[21] A. Kassam, L. Trefethen, Fourth-order time stepping for stiff PDEs., SIAM J. Sci. Comput. 26 (4) (2005) 1214–1233.
[22] S. Krogstad, Generalized integrating factor methods for stiff PDEs, J. Comp. Phys. 203 (2005) 72–88.
[23] R. Lefever, G. Nicolis, Chemical instabilities and sustained oscillations., J. Theor. Biol. 3 (1971) 267–284.
[24] P. Gray, S. K. Scott, Autocatalytic reactions in the isothermal continuous stirred tank reactor., Chem. Engng. Sci. 39 (1984) 1087–1097.
[25] B. Schmitt, R. Weiner, Matrix-free w-methods using a multiple arnoldi iteration., Appl. Numer. Math. 18 (1995) 307–320.